Biometrika (2022), **xx**, x, *pp*. 1–20
Printed in Great Britain

Gradient-based Sparse Principal Component Analysis with Extensions to Online Learning

BY YIXUAN QIU

Department of Statistics and Data Science, Carnegie Mellon University, Pittsburgh, U.S.A
School of Statistics and Management, Shanghai University of Finance and Economics, Shanghai, China
qiuyixuan@sufe.edu.cn

JING LEI

Department of Statistics and Data Science, Carnegie Mellon University, Pittsburgh, U.S.A
jinglei@andrew.cmu.edu

AND KATHRYN ROEDER

Department of Statistics and Data Science, Carnegie Mellon University, Pittsburgh, U.S.A
roeder@andrew.cmu.edu

SUMMARY

Sparse principal component analysis (PCA) is an important technique for simultaneous dimensionality reduction and variable selection of high-dimensional data. In this work we combine the unique geometric structures of the sparse PCA problem with recent advances in convex optimization to develop novel gradient-based sparse PCA algorithms. These new algorithms enjoy the same global convergence guarantee as the original alternating direction method of multipliers, and can be more efficiently implemented with the rich toolbox developed for gradient methods in the deep learning literature. Most notably, these gradient-based algorithms can be combined with stochastic gradient descent methods, leading to efficient online sparse PCA algorithms with provable numerical and statistical performance guarantees. The practical performance and usefulness of the new algorithms are demonstrated in various simulation studies. As an application, the scalability and statistical accuracy of our method enable us to find interesting functional gene groups in high-dimensional RNA sequencing data.

Some key words: Sparse principal component analysis, Dimensionality reduction, Convex optimization, Gradient descent, Online learning

1. INTRODUCTION

Principal component analysis (PCA, Pearson, 1901; Hotelling, 1933) is a classical yet indispensable dimensionality reduction technique in statistics and machine learning. PCA generates higher-level features of the raw data by computing uncorrelated linear combinations of the original variables that retain the maximum amount of variation of the raw data. See Jolliffe (2002) for a more detailed introduction and applications.

In high-dimensional settings, where the number of variables can be comparable to or larger than the sample size, standard PCA may suffer from the curse-of-dimensionality. For instance,

Johnstone & Lu (2009) and Jung & Marron (2009) showed that when the number of variables is much larger than the sample size, PCA can behave poorly in estimating the principal components even with a simple population covariance structure, producing misleading results. These theoretical works motivated the sparse PCA method, which overcame many of the limitations of standard PCA in high-dimensional settings. Sparse PCA works similarly to standard PCA, but requires the principal components to be sparse, so that they only involve a few original variables in the linear combinations. Such a sparsity requirement greatly reduces the parameter space, and enhances the interpretability of the estimated principal components. Pioneering work on sparse PCA includes Jolliffe et al. (2003); Johnstone & Lu (2009) and Zou et al. (2006). Since then sparse PCA has found wide applications in keyword extraction for text data (Zhang & Ghaoui, 2011), fault detection for industrial processes (Grbovic et al., 2012; Gajjar et al., 2018), genomics and genetics (Lee et al., 2012; Zhu et al., 2017), among many others.

Despite the theoretical advantage of sparse PCA, its practical application remains a challenge, especially in modern large-scale data sets. The main bottleneck is that sparse PCA involves solving a nonconvex optimization problem (Jolliffe et al., 2003). Existing fast algorithms using nonconvex objective functions (Zou et al., 2006; Witten et al., 2009; Journée et al., 2010) generally do not guarantee global convergence and are sensitive to the initial values, or require special structures on the true covariance matrix (Ma, 2013). Convex relaxation methods (d'Aspremont et al., 2005; Vu et al., 2013) are guaranteed to converge globally and have desirable statistical properties, but rely on semidefinite programming, which is computationally expensive for large input covariance matrices.

Another major challenge in using sparse PCA in practice is the limited development of online algorithms. Online algorithms, such as the celebrated stochastic gradient descent method, provide the most powerful and efficient framework for large-scale optimization problems. While online PCA algorithms have been extensively studied in the recent literature (Oja & Karhunen, 1985; Warmuth & Kuzmin, 2008; Marinov et al., 2018; Li et al., 2018), online sparse PCA has seen much less progress (Yang & Xu, 2015; Wang & Lu, 2016). The difficulty is that existing methods could not express sparse PCA as an easy-to-solve unconstrained or trivially constrained optimization problem, due to the complex structure of the constraint set.

In this article we improve the applicability and computational efficiency of sparse PCA in two important ways. First, we develop a gradient-based algorithm that solves the same sparse PCA problem with convex relaxation (d'Aspremont et al., 2005; Vu et al., 2013). The new algorithm can be implemented with cutting-edge tools developed for gradient descent methods, and hence enjoys superior computational efficiency than the original alternating direction method of multipliers (Boyd et al., 2011), especially for large-scale problems. Second, we further extend the gradient-based algorithm to the online setting, where variants of stochastic gradient descent can be applied. To our best knowledge, this is the first online sparse PCA algorithm that can be computed efficiently in a genuine online fashion without diverging minibatch sizes, and has global convergence as well as statistical estimation accuracy guarantees for a general covariance model.

At the core of our algorithmic development is a novel and profound understanding of the geometry of the sparse PCA problem. Roughly speaking, the main challenge in solving the sparse PCA problem, even after convex relaxation, is the complexity of the constraint set, which involves the intersection of three convex bodies in the space of symmetric matrices. In order to transform such a constrained problem to a trivially constrained problem using recent advances in convex optimization (Kundu et al., 2018; Mahdavi et al., 2012; Yang et al., 2017), a key intermediate step is to understand the relationship between the projection operators of these individual convex bodies and the projector of their intersection. The convex constraint set in the PCA problem is called the *Fantope*, which is the convex hull of all projection matrices of a certain rank.

To solve this problem, our main theoretical result, Theorem 2, uses an analytic representation of the Fantope to derive an upper bound of its projection distance in terms of simpler projections.

The practical implementation and performance of the proposed algorithms are demonstrated by various simulation experiments. For the batch version sparse PCA, our simulation shows that the new algorithm converges much faster than the alternating direction method of multipliers. In online settings, the proposed algorithms also have convergence properties prescribed by the theory. Moreover, we apply the new sparse PCA algorithm to a real high-dimensional gene expression data set, successfully detecting differential co-expression patterns in schizophrenia subjects compared to a control group. Our core algorithm is implemented in the `gradfps` R package available at `https://github.com/yixuan/gradfps`, and the code to reproduce the results in this paper is provided in the supplementary material.

2. PRELIMINARIES

Suppose the data set is a sample of independent and identically distributed random vectors $Z_1, \ldots, Z_n \in \mathbb{R}^p$ with zero means and population covariance matrix $\Sigma = \mathbb{E}(Z_1 Z_1^{\mathrm{T}})$. Let $(\theta_j, \gamma_j)_{j=1}^p$ be the eigenvalue-eigenvector pairs of Σ such that $\theta_1 \geq \cdots \geq \theta_p \geq 0$. The PCA problem is concerned with estimating the top d eigenvectors of Σ for some small positive integer d. In the high-dimensional setting where p is comparable to or larger than n, estimating the principal components can be statistically hard, and hence we consider the following sparsity condition.

Assumption 1. The factor loading matrix $\Gamma = [\gamma_1, \ldots, \gamma_d]$ has at most s nonzero rows, and the dth eigen-gap of Σ is nonzero, $\delta_d = \theta_d - \theta_{d+1} > 0$.

Such a sparsity assumption is called the "row sparsity" in Vu & Lei (2013), and facilitates both theoretical understanding and practical interpretation. In high-dimensional problems such as genetics, a sparse factor loading matrix provides simultaneous dimension reduction and variable selection. The eigen-gap condition is required so that subspace spanned by the top d eigenvectors is uniquely defined.

Following the sparsity assumption, sparse PCA estimators have been derived in many different ways, including the lasso approach (Jolliffe et al., 2003), regression-based formulation (Zou et al., 2006), iterative thresholding methods (Shen & Huang, 2008; Witten et al., 2009; She, 2017), the generalized power method (Journée et al., 2010), among many others. Also see Zou & Xue (2018) for a recent review of various sparse PCA methods. These algorithms are typically solving nonconvex optimization problems, which possess local convergence at best, unless additional structural assumptions are made on Σ (Ma, 2013).

In this paper we focus on another class of sparse PCA methods that use convex relaxation with guaranteed global convergence in polynomial time (d'Aspremont et al., 2005; Vu et al., 2013). Let $S = n^{-1} \sum_{i=1}^n Z_i Z_i^{\mathrm{T}}$ be the empirical covariance matrix, and X be a rank-d projection matrix in $\mathbb{R}^{p \times p}$. Then the total variance in the d-dimensional subspace corresponding to X is $\mathrm{tr}(\Sigma X)$, with empirical version $\mathrm{tr}(SX)$, where $\mathrm{tr}(\cdot)$ is the trace of a matrix. The convex relaxation sparse PCA adds to the total variance a lasso-type penalty:

$$\begin{aligned} \max_{X} \quad & \mathrm{tr}(SX) - \lambda \|X\|_{1,1} \\ \text{s.t.} \quad & O \preceq X \preceq I \ \text{ and } \ \mathrm{tr}(X) = d, \end{aligned} \tag{1}$$

where λ is the sparsity penalty parameter, $\|A\|_{p,q} = \{\sum_{j=1}^n (\sum_{i=1}^m |a_{ij}|^p)^{q/p}\}^{1/q}$ stands for the $L_{p,q}$ norm for an $m \times n$ matrix A, O and I denote the zero and identity matrices, respectively, and $A \preceq B$ means $B - A$ is nonnegative definite. The convex feasible set $\mathcal{F}^d = \{X : O \preceq X \preceq I \ \text{ and } \ \mathrm{tr}(X) = d\}$, called the Fantope, is the convex hull of all rank-d projection matrices. Prob-

lem (1) is called Fantope projection and selection (Vu et al., 2013). When $d = 1$, it becomes equivalent to the direct formulation for sparse PCA proposed in d'Aspremont et al. (2005). The formulation (1) has attractive statistical properties (Vu et al., 2013; Lei & Vu, 2015), and can be solved in polynomial time using alternating direction method of multipliers.

However, the existing algorithm is slow when p is moderately large (a few hundreds), since each iteration requires projecting a $p \times p$ matrix onto the Fantope, which involves a full eigen-decomposition of the $p \times p$ matrix. As a consequence, the applicability of Fantope projection and selection is substantially limited by the cubic growth of computing time per iteration.

3. A NEARLY PROJECTION-FREE CONVEX RELAXATION FOR SPARSE PCA

3.1. Optimization on Intersection of Convex Sets

Our approach to developing an efficient sparse PCA algorithm is based on converting the constrained problem (1) to an equivalent but trivially constrained one, so that we can exploit many existing powerful tools from gradient-based unconstrained convex optimization. Here by "trivially constrained" we mean the constraint has a simple form and the projection on to the constraint set is easy to compute. The main challenge in applying the gradient based methods to problem (1) stems from the constraint set, which is the intersection of three simpler convex sets: $\mathcal{F}_1 = \{X : \mathrm{tr}(X) = d\}$, $\mathcal{F}_2 = \{X : X \succeq O\}$, and $\mathcal{F}_3 = \{X : X \preceq I\}$. While each one of the three sets has a simple structure, the associated projection operator of the intersection becomes the major obstacle for an efficient algorithm.

Our main strategy is to show that, under certain assumptions, the complex constraint can be replaced by adding an appropriately calibrated penalty term to the objective function, resulting in an equivalent problem with only a trivial constraint. Unlike the method of Lagrange multiplier, the penalty term in the equivalent formulation is fixed and does not involve new auxiliary variables. Moreover, the penalty term is decomposable with respect to the individual constraint sets, significantly reducing the computational burden.

To this end, consider a general optimization problem of the following abstract form:

$$\min_{x \in \mathcal{K}} f(x), \quad \mathcal{K} = C_1 \cap \cdots \cap C_l \cap G_1 \cap \cdots \cap G_m, \tag{2}$$

where $f(x)$ is a convex objective function defined on an Euclidean space, C_i's are closed convex sets, and each $G_i = \{x : g_i(x) \leq 0\}$ is defined by a convex function $g_i(x)$. We further assume that $\mathcal{K}$ is contained in a closed convex set $\mathcal{X} \subset \mathbb{R}^p$ whose projection operator $\mathcal{P}_{\mathcal{X}}$ is trivial, where $\mathcal{P}_C(x) = \arg\min_{y \in C} \|y - x\|$, with $\|\cdot\|$ being the Euclidean norm. The intersection set $\mathcal{K}$ is decomposed in such a way that the projection operators $\mathcal{P}_{C_i}$ and the constraint functions $g_i(x)$ are easy to compute.

A special case of problem (2) with $m = 0$ has been studied in the literature (Kundu et al., 2018). Another special case with $l = 0$ and $m = 1$ has been studied in Mahdavi et al. (2012) and Yang et al. (2017). However, none of these special cases cover problem (1), which corresponds to the case of $l = 1$ and $m = 2$. To the best of our knowledge, the general case with both l, $m > 0$ has not been studied in the literature.

We make the following assumptions on the objects involved in (2).

Assumption 2. $f(x)$ is Lipschitz continuous on $\mathcal{X}$ with Lipschitz constant $L > 0$: $|f(x) - f(y)| \leq L\|x - y\|, \forall x, y \in \mathcal{X}$.

Assumption 3. For $i = 1, \ldots, m$, (a) $x \in \mathcal{X}$ implies $\mathcal{P}_{G_i}(x) \in \mathcal{X}$; (b) there exists a constant ρ_i such that $\inf_{v \in \mathcal{D}_i} \|v\| \geq \rho_i > 0$, where $\mathcal{D}_i = \{v : v \in \partial g_i(x), g_i(x) = 0, x \in \mathcal{X}\}$, and $\partial g_i(x) = \{v : g_i(y) - g_i(x) \geq v^{\mathrm{T}}(y - x), \forall y\}$ is the subdifferential of g_i at x.

Assumption 4. There exists a multivariate function $h : [0, +\infty)^{l+m} \mapsto [0, +\infty)$ such that (a) $h(0, \ldots, 0) = 0$, (b) h is non-decreasing in each argument, and (c) for all $x \in \mathcal{X}$,

$$d_{\mathcal{K}}(x) \leq h\left(d_{C_1}(x), \ldots, d_{C_l}(x), d_{G_1}(x), \ldots, d_{G_m}(x)\right), \tag{3}$$

where $d_C(x) = \|x - \mathcal{P}_C(x)\|$ is the distance between x and C.

Assumption 2 is a common smoothness condition for objective functions. Assumption 3 is derived from Yang et al. (2017), and can also be easily verified given concrete $g_i(x)$ functions. Assumption 4 is the most non-trivial one and is the key to removing the constraints in problem (2). It reflects the geometric features of the constraint set and often requires case-by-case analysis. Verifying Assumption 4 for problem (1) is a major technical contribution of this paper, and is the focus of Section 3.2 below.

Define the function

$$\mathcal{L}(x; \mu) = f(x) + \mu h\left(d_{C_1}(x), \ldots, d_{C_l}(x), \rho_1^{-1}[g_1(x)]_+, \ldots, \rho_m^{-1}[g_m(x)]_+\right),$$

where $[x]_+ = \max\{x, 0\}$. Then the following theorem, which can be seen as a generalization to Proposition 2 of Kundu et al. (2018), states the equivalence between (2) and a trivially constrained optimization problem.

THEOREM 1. *Suppose that Assumptions 2 to 4 hold, and define* $f_* = \min_{x \in \mathcal{K}} f(x)$ *and* $\mathcal{L}_* = \min_{x \in \mathcal{X}} \mathcal{L}(x; \mu)$. *Let* $x_\varepsilon \in \mathcal{X}$ *be an approximate solution such that* $\mathcal{L}(x_\varepsilon; \mu) \leq \mathcal{L}_* + \varepsilon$ *for* $\varepsilon > 0$, *and* $y_\varepsilon = \mathcal{P}_{\mathcal{K}}(x_\varepsilon)$. *Then the following results hold: (a) If* $\mu \geq L$, *then* $f_* = \mathcal{L}_*$; *(b) If* $\mu \geq L + 1$, *then* $\|x_\varepsilon - y_\varepsilon\| \leq \varepsilon$, $\mathcal{L}(y_\varepsilon; \mu) \leq \mathcal{L}_* + \varepsilon$, *and* $f(y_\varepsilon) \leq f_* + \varepsilon$.

Although the alternative problem $\min_{x \in \mathcal{X}} \mathcal{L}(x; \mu)$ still involves a constraint set $\mathcal{X}$, it is much different from the original constrained one, as we require the projection operator $\mathcal{P}_{\mathcal{X}}$ to be trivial. Therefore, optimizing $\mathcal{L}(x; \mu)$ over $\mathcal{X}$ is a trivially constrained problem.

3.2. The Gradient FPS Algorithm

For clarity we first define the following notations:

$$\begin{aligned}
\mathbb{S} &= \{X \in \mathbb{R}^{p \times p} : X = X^{\mathrm{T}}\}, & \mathcal{X} &= \{X \in \mathbb{S} : \|X\|_F^2 \leq d\}, \\
\mathcal{F}_1 &= \{X \in \mathbb{S} : \mathrm{tr}(X) = d\}, & \mathcal{F}_{2,3} &= \{X \in \mathbb{S} : O \preceq X \preceq I\}, \\
G_1 &= \{X \in \mathbb{S} : g_1(X) \leq 0\}, & g_1(X) &= \theta_1(X) - 1, \\
G_2 &= \{X \in \mathbb{S} : g_2(X) \leq 0\}, & g_2(X) &= -\theta_p(X), \\
\mathcal{K} &= \mathcal{F}_1 \cap G_1 \cap G_2, & f(X) &= -\mathrm{tr}(SX) + \lambda \|X\|_{1,1},
\end{aligned}$$

where $\|\cdot\|_F$ is the Frobenius norm and $\theta_j(\cdot)$ is the jth largest eigenvalue. Then it is easy to find that problem (1) can be written in the form of (2) with $l = 1, m = 2, C_1 = \mathcal{F}_1$, and the f, g_1, g_2 functions defined above.

To apply Theorem 1, we need to verify Assumptions 2 to 4, among which Assumption 4 plays a central role. The following theorem, which describes the geometry of the Fantope, is the key to validating this assumption.

THEOREM 2. *For any* $1 \leq d \leq (p-1)/2$ *and any* $p \times p$ *symmetric matrix* X,

$$d_{\mathcal{F}^d}(X) \leq (\sqrt{2} + 1) p^{1/2} \left\{(d+1)^{-1/2} \cdot d_{\mathcal{F}_1}(X) + d_{\mathcal{F}_{2,3}}(X)\right\}. \tag{4}$$

Theorem 2 is proved using the properties of normal cones of convex polytopes, with a combination of analytical and geometrical arguments. With inequality (4), we are able to verify the required assumptions in the following corollary.

COROLLARY 1. *For problem* (1), *if* $d \leq (p-1)/2$, *then*

1. *$f(X)$ satisfies Assumption 2 with some $L \le \|S\|_F + \lambda p$.*
2. *Assumption 3 holds with $\rho_1 = d^{-1/2}$ and $\rho_2 = p^{-1/2}$.*
3. *$d_{\mathcal{K}}(X) \le (\sqrt{2}+1)p^{1/2}\left\{(d+1)^{-1/2} \cdot d_{C_1}(X) + d_{G_1}(X) + d_{G_2}(X)\right\}$.*

As a consequence, define

$$\mathcal{L}(X) = -\mathrm{tr}(SX) + \lambda\|X\|_{1,1} + \mu\left(d_{C_1}(X) + r_1[g_1(X)]_+ + r_2[g_2(X)]_+\right), \tag{5}$$

and then for any $\mu \ge (\sqrt{2}+1)(L+1)\{p/(d+1)\}^{1/2}$, $r_1 \ge \{d(d+1)\}^{1/2}$, and $r_2 \ge \{p(d+1)\}^{1/2}$, we have $\min_{X\in\mathcal{K}} f(X) = \min_{X\in\mathcal{X}} \mathcal{L}(X)$.

Corollary 1 is significant as it opens the door to a large collection of tools to solve the trivially constrained problem (5), as a substitute for the once highly constrained and difficult form (1). For illustrative purpose, we consider two algorithms to solve (5) as some natural and familiar choices. We expect that more advanced and efficient methods can also be used, such as various optimization techniques broadly used in the deep learning community (Duchi et al., 2011; Zeiler, 2012; Kingma & Ba, 2015; Luo et al., 2019).

In what follows, $\mathcal{S}_\alpha(x) = \mathrm{sign}(x)\cdot\max\{|x|-\alpha, 0\}$ is the soft-thresholding operator, and for a matrix argument X, $\mathcal{S}_\alpha(X)$ means applying $\mathcal{S}_\alpha$ to X elementwise. The proximal operator of a convex function f is defined as $\mathrm{prox}_{\alpha f}(x) = \arg\min_u \left\{f(u) + (2\alpha)^{-1}\|u-x\|^2\right\}$, where α is the step size.

Algorithm 1: Subgradient method. The first and most straightforward algorithm is the subgradient descent method, which, at step k, performs the update $X_{k+1} = \mathcal{P}_{\mathcal{X}}\left(X_k - \alpha_k \nabla\mathcal{L}(X_k)\right)$, where α_k is the step size and $\nabla\mathcal{L}(X_k)$ is the subgradient of $\mathcal{L}(X)$ at X_k. The eigenvalues and eigenvectors are computed using the implicitly restarted Lanczos method (Sorensen, 1997), which can be viewed as an improved power method (Warsa et al., 2004). Algorithm 1 is a variant of the conventional subgradient descent called incremental proximal method (Bertsekas, 2011). When the step sizes α_k are at the order of $\mathcal{O}(k^{-1/2})$, Algorithm 1 has a convergence rate of $\mathcal{O}(T^{-1/2})$ on the optimization error, where T is the number of iterations. This rate is slower than that of Algorithm 2 introduced below, but it is computationally efficient, and is typically used in the early exploration of solutions. More importantly, we show in Section 4.2 that the subgradient method is especially powerful in dealing with streaming data.

Algorithm 2: Proximal-proximal-gradient method. The second algorithm, which has a better convergence rate of $\mathcal{O}(T^{-1})$, is the proximal-proximal-gradient method (Ryu & Yin, 2019) given in Algorithm 2, where

$$\begin{aligned} &f_1(X) = \lambda\|X\|_{1,1}, && \mathrm{prox}_{\alpha f_1}(X) = \mathcal{S}_{\alpha\lambda}(X), \\ &f_2(X) = -\mathrm{tr}(SX) + \mu d_{C_1}(X) + \\ &\qquad \mu r_1[g_1(X)]_+ + \mu r_2[g_2(X)]_+, && \mathrm{prox}_{\alpha f_2}(X) : \text{Appendix A.3.} \end{aligned}$$

The computation of $\mathrm{prox}_{\alpha f_2}(X)$, with details in Appendix A.3, requires retrieving the leading eigenvalues of a dense matrix and the associated eigenvectors, possibly more than one. In each iteration, Algorithm 2 is faster than a full eigen-decomposition as in the existing alternating direction method of multipliers for (1).

Remark 1. In practice we recommend using a hybrid approach to combine the best of both algorithms: we first run the fast subgradient method a few iterations to rapidly decrease the objective function value in the early stage, and then use Algorithm 2 for fine-tuning. We term this method, given in Algorithm 3, as the gradient-based sparse PCA. Also, since the subgradient method is only for computing the warm start, its step size sequence $\{\alpha_k\}$ does not need to strictly follow the $\mathcal{O}(k^{-1/2})$ order, and can take larger values in practice.

Algorithm 1: The subgradient algorithm to solve (5)

Input: $S, T, \{\alpha_k\}$, initial value $X_0 \in \mathcal{X}$
Output: $\hat{X}$

for $k = 1, \ldots, T$ *do*
  $X_k^{(0)} \leftarrow X_{k-1}$
  $X_k^{(1)} \leftarrow \mathcal{S}_{\alpha_k \lambda}(X_k^{(0)})$
  $X_k^{(2)} \leftarrow X_k^{(1)} - \alpha_k \mu r_1 \mathbf{1}\{\theta_1 > 1\}\gamma_1\gamma_1^{\mathrm{T}} + \alpha_k \mu r_2 \mathbf{1}\{\theta_p < 0\}\gamma_p\gamma_p^{\mathrm{T}}$,
  where $\theta_i = \theta_i(X_k^{(1)})$, $\gamma_i = \gamma_i(X_k^{(1)})$, $i = \{1, p\}$
  $X_k^{(3)} \leftarrow X_k^{(2)} + \min\{\beta, 1\} \cdot s \cdot I$, where $s = \{d - \mathrm{tr}(X_k^{(2)})\}/p$, $\beta = \alpha_k \mu / (p^{1/2}|s|)$
  $X_k \leftarrow \mathcal{P}_{\mathcal{X}}\left(X_k^{(3)} + \alpha_k S\right) = \min\left\{1, d^{1/2}/\|X_k^{(3)} + \alpha_k S\|_F\right\} \cdot \left(X_k^{(3)} + \alpha_k S\right)$
end for
return $\hat{X} = T^{-1} \sum_{k=1}^{T} X_k$

Algorithm 2: The proximal-proximal-gradient algorithm to solve (5)

Input: S, T, α, initial value $X_0 \in \mathcal{X}$
Output: $\hat{X}$

Set $Z_0^{(1)} = Z_0^{(2)} \leftarrow X_0$
for $k = 0, 1, \ldots, T-1$ *do*
  $\bar{Z}_k \leftarrow (Z_k^{(1)} + Z_k^{(2)})/2$
  $X_{k+1} \leftarrow \mathcal{P}_{\mathcal{X}}\left(\bar{Z}_k\right) = \min\left\{1, d^{1/2}/\|\bar{Z}_k\|_F\right\} \cdot \bar{Z}_k$
  $Z_{k+1}^{(1)} \leftarrow Z_k^{(1)} - X_{k+1} + \mathrm{prox}_{\alpha f_1}(2X_{k+1} - Z_k^{(1)})$
  $Z_{k+1}^{(2)} \leftarrow Z_k^{(2)} - X_{k+1} + \mathrm{prox}_{\alpha f_2}(2X_{k+1} - Z_k^{(2)})$
end for
return $\hat{X} = T^{-1} \sum_{k=1}^{T} X_k$

Algorithm 3: The gradient-based sparse PCA

Input: $S, B, T, \{\alpha_k\}, \alpha$, initial value $\tilde{X}_0 \in \mathcal{X}$
Output: $\hat{X}$

$X_0 \leftarrow$ `subgradient_algorithm`$(S, B, \{\alpha_k\}, \tilde{X}_0)$, given by Algorithm 1
$\hat{X} \leftarrow$ `ppg_algorithm`(S, T, α, X_0), given by Algorithm 2

Remark 2. The theoretical upper bound of the constant μ provided in Corollary 1 may be very large, especially when p is large. This is a conservative choice to make the upper bound argument in Theorem 1 valid for all $x \in \mathcal{X}$. In practice there is no reason to stick with such a conservative choice. For most "typical" regions in the space $\mathcal{X}$, the penalty parameter can take much smaller values. See Appendix A.1 for an illustration of this point. Inspired by Lepskii's method of selecting tuning parameters (Lepskii, 1991), we use the following scheme in the actual implementation: fix a small value $\mu = \mu_0$, and compute the solution $\hat{X}$; then double the size of μ with $\mu = 2\mu_0$ and obtain the new solution $\hat{X}'$. If $\|\hat{X}' - \hat{X}\|_F < \varepsilon$ for some tolerance ε, then accept $\hat{X}'$ as the final solution; otherwise repeat this procedure until the solution is stable under doubling the value of μ.

Remark 3. One may also wonder about applying the proximal-proximal-gradient algorithm directly to problem (1) with the constraints represented by convex indicators in the objective function. Indeed, this method may result in different algorithms to solve (1) by appropriately decomposing the objective function into simpler components. However, a notable difference between this version and (5) is that the new objective function in (5) is always finite, even for infeasible iterates; more importantly, it is also Lipschitz continuous, thus leading to a sublinear convergence speed as demonstrated in Theorem 3. In contrast, the non-differentiability and the possible infinite value of the indicators will break such theoretical guarantee. For example, it is possible that the iterates are always infeasible for some constraints, and hence the objective function value stays at ∞. This unfavorable property makes it hard to track the optimization progress and test convergence.

3.3. Convergence Analysis

One remarkable benefit of the gradient-based algorithm is that we can bound its optimization error at any finite iteration step. With a sufficiently large number of iterations, Algorithm 3 can be shown to output an approximate solution with an arbitrary precision. Since the subgradient descent stage is only used for the warm start, without loss of generality we assume $B = 0$ in Algorithm 3. Theorem 3 below provides an explicit upper bound for the optimization error.

THEOREM 3. *The output $\hat{X}$ of Algorithm 3 satisfies*

$$\mathcal{L}(\hat{X}) \leq \min_{X \in \mathcal{X}} \mathcal{L}(X) + \frac{C}{T} \quad \textit{and} \quad d_{\mathcal{K}}(\hat{X}) \leq \frac{C}{T},$$

where $C = C(S, X_0, \alpha, \lambda, p, d)$ is a constant free of T, with the explicit expression given in Appendix A.2.

Remark 4. Unlike common statistical estimation error bounds, Theorem 3 provides an *optimization error* of the algorithm, which bounds the difference between the theoretical global optimum and the finite step solution. Here the optimization problem is treated as fixed and we seek a good dependence on T, the number of iterations. The constant C in the statement of the theorem (see also in Corollary 2 and Theorem 4) may depend on the optimization problem, which involves p. However, as we noted in Remark 2, in practice the actual dependence of C on p is usually much better than the worst case.

If the optimization problem $\min_{X \in \mathcal{X}} \mathcal{L}(X)$ is solved exactly with the solution $\hat{X}_*$, then the statistical property of $\hat{X}_*$ has already been studied by Vu et al. (2013). In practice, only a finite-precision solution $\hat{X}$ can be obtained, which usually does not exactly minimize the objective function, and is not in the constraint set $\mathcal{K}$. Corollary 2 below ensures that $\hat{X}$ is still a good estimator for the principal subspace projector $\Pi = \Gamma\Gamma^{\mathrm{T}}$, with an explicit upper bound for the estimation error as a function of the sample size n and the number of iterations T.

Assumption 5. There exists a constant $\sigma > 0$ such that $\max_{i,j} P(|S_{ij} - \Sigma_{ij}| \geq u) \leq 2\exp(-4nu^2/\sigma^2)$ for all $u \leq \sigma$.

Assumption 5 has been used in the sparse PCA literature before Vu et al. (2013), and holds in many prototypical settings such as when the random vector Z_1 is σ-sub-Gaussian.

COROLLARY 2 (ESTIMATION ERROR BOUND). *Suppose that Assumptions 1 and 5 hold, and take $\lambda = \sigma\{\log(p)/n\}^{1/2}$. Then with probability at least $1 - 2/p^2$, we have*

$$\|\hat{X} - \Pi\|_F \leq \frac{4\sigma s\{\log(p)\}^{1/2}}{\delta_d n^{1/2}} + \frac{(2C/\delta_d)^{1/2}}{T^{1/2}} + \frac{C}{T}, \tag{6}$$

where C is given in Theorem 3.

If a rank-d projector is wanted, let $\hat{\Pi}$ be the leading rank-d projector of $\hat{X}$. Using triangle inequality $\|\hat{\Pi} - \Pi\|_F \leq \|\hat{\Pi} - \hat{X}\|_F + \|\hat{X} - \Pi\|_F$ and the fact that $\|\hat{\Pi} - \hat{X}\|_F \leq \|\Pi - \hat{X}\|_F$, we have $\|\hat{\Pi} - \Pi\|_F \leq 2\|\hat{X} - \Pi\|_F$ so that the estimation error $\|\hat{\Pi} - \Pi\|_F$ is upper bounded by twice the right hand side of (6) with a high probability.

The error bound (6) has an intuitive interpretation. The first term quantifies the *statistical error*, which depends on the $\log(p)$ term that is common in high-dimensional data analysis. The second term is the *optimization error*, which decays at the $\mathcal{O}(T^{-1/2})$ rate. The last term is the *feasibility error*, since $\hat{X}$ is not necessarily a projection matrix.

4. Online Sparse PCA

4.1. Online Learning Setting

In this section we consider the scenario in which data are obtained in a streaming fashion. Streaming data reflect many practical needs that data acquisition and computation happen roughly at the same time. For instance, the complete data collection procedure may span a long period of time, or the data set is too large to be stored entirely on the machine. In both cases, it is desirable to make full use of the existing data, and then update the model parameters when new data points come in. Such algorithms are typically called online learning algorithms. Correspondingly, the algorithms that use the whole data set, for instance Algorithm 3, are referred to as offline learning or batch learning algorithms.

Formally, we assume that there is an infinite sequence of independent random vectors $Z_1, Z_2, \ldots \in \mathbb{R}^p$ with $\mathbb{E}(Z_t) = 0$ and $\mathrm{cov}(Z_t) = \mathbb{E}(S_t) = \Sigma$, $t \geq 1$, where $S_t = Z_t Z_t^{\mathrm{T}}$. We consider the same sparsity assumption (Assumption 1) for the population covariance matrix Σ, and the estimation target is the top-d projection matrix Π of Σ. We define the online learning procedure as follows. At each time point t, the data analyst constructs an estimator X_t for Π. To match the nature of streaming data, we require that X_t only depends on Z_t, X_{t-1}, and optionally some other quantities that depend on the history $\{Z_i\}_{i=0}^t$ with a storage size not growing with t. The procedure stops at time T, and a final estimator $\hat{X}_T$ is output by the online learning algorithm. For clarity, T is also called the sample size of the streaming data in this context.

After each X_t is constructed, we use it to predict the next data point Z_{t+1} with loss function

$$\ell_t(X_t, Z_{t+1}) = -Z_{t+1}^{\mathrm{T}} X_t Z_{t+1} + \lambda \|X_t\|_{1,1} + \nu d_{\mathcal{K}}(X_t), \tag{7}$$

where λ and ν are constants. In this loss function, the first term quantifies the (negative) explained variance on new data if X_t is treated as a projection matrix, the second term encourages the sparsity of X_t, and the third term penalizes the deviation from the constraint set $\mathcal{K} = \mathcal{F}^d$. For the whole sequence, define the total excess loss

$$\mathcal{R}(\{X_t\}, \{Z_t\}, T) = \sum_{t=1}^{T} \ell_t(X_t, Z_{t+1}) - \sum_{t=1}^{T} \ell_t(\Pi, Z_{t+1})\,. \tag{8}$$

Naturally, a good online learning algorithm should have a strict control of the regret, defined as the expected total excess loss, as a function of T.

Unlike most online optimization algorithm studies, our analysis also covers the statistical estimation error of the final output of the online learning algorithm, measured by the quantity $\|\hat{X}_T - \Pi\|_F$. In our setting, the final output $\hat{X}_T$ depends on the data stream in a complicated way, as the model is updated using both historical and new information, making the model increment $X_t - X_{t-1}$ correlated across iterations. Therefore, bounding the statistical error of $\hat{X}_T$ becomes a non-trivial yet important problem in online estimation scenarios.

Algorithm 4: The online gradient-based sparse PCA

Input: $\{Z_t\}$, T, $\{\alpha_t\}$, initial value X_0

Output: $\hat{X}_T$

for $t = 1, \ldots, T$ *do*

$X_t^{(0)} \leftarrow X_{t-1}$

$X_t^{(1)} \leftarrow \mathcal{S}_{\alpha_t \lambda}(X_t^{(0)})$

$X_t^{(2)} \leftarrow X_t^{(1)} - \alpha_t \nu_0 (pd)^{1/2} \mathbf{1}\{\theta_1 > 1\} \gamma_1 \gamma_1^{\mathrm{T}} + \alpha_t \nu_0 p \mathbf{1}\{\theta_p < 0\} \gamma_p \gamma_p^{\mathrm{T}}$,
where $\nu_0 = (\sqrt{2}+1)\nu$, $\theta_i = \theta_i(X_t^{(1)})$, $\gamma_i = \gamma_i(X_t^{(1)})$, $i = \{1, p\}$

$X_t^{(3)} \leftarrow X_t^{(2)} + \min\{\beta, 1\} \cdot s \cdot I$, where $s = (d - \mathrm{tr}(X_t^{(2)}))/p$, $\beta = \alpha_t \nu_0 / \{(d+1)|s|\}$

$X_t \leftarrow \mathcal{P}_{\mathcal{X}} \left(X_t^{(3)} + \alpha_t S_t \right) = \min \left\{ 1, d^{1/2} / \| X_t^{(3)} + \alpha_t S_t \|_F \right\} \cdot \left(X_t^{(3)} + \alpha_t S_t \right)$

end for

return $\hat{X}_T = T^{-1} \sum_{t=1}^{T} X_t$

4.2. The Online Gradient-based Sparse PCA

Under the setting in Section 4.1, we propose a gradient-based algorithm to solve the streaming sparse PCA problem, aiming at efficient iterations with provable control of the regret and statistical estimation error. Given the nature of streaming data, our development will focus on the case of large T, while treating the dimension p as a fixed number.

Thanks to the trivially constrained form of the objective function in Corollary 1, the stochastic subgradient method can be used to compute sparse PCA on streaming data. The proposed online gradient-based sparse PCA, outlined in Algorithm 4, is the online version of Algorithm 1. The only non-trivial step in Algorithm 4 is to compute the eigenvalues of $X_t^{(1)}$, which is a sparse matrix as a result of the soft-thresholding operator. Computing the largest and smallest eigenvalues of a sparse matrix is much more efficient than the full eigen-decomposition of a dense matrix, since its complexity mostly depends on the number of nonzero elements. As a result, the per-iteration cost of Algorithm 4 is very small.

Besides the computational advantage, the following theorem shows that if $\|S_t\|_F$ is properly bounded, then the average regret of Algorithm 4 decays at the rate of $\mathcal{O}(T^{-1/2})$, which matches the best known result for the online subgradient method with a non-strongly-convex objective function (Abernethy et al., 2008).

Assumption 6. The data sequence $\{Z_t\}$ can be expressed as $Z_t = RU_t$, where R is a $p \times p$ matrix, and $\{U_t\}$ is an infinite sequence of independent random vectors. Each U_t has independent components $U_{t,1}, \ldots, U_{t,p}$, where $U_{t,i}$ is a zero-mean sub-Gaussian random variable, *i.e.*, $\mathbb{E}(e^{\lambda U_{t,i}}) \le e^{\lambda^2/2}$ for all $\lambda \in \mathbb{R}$.

THEOREM 4. *Let* $\alpha_1 = \alpha_0 > 0$ *and* $\alpha_t = \alpha_0 (t-1)^{-1/2}$ *for* $t \ge 2$. *Then there exist constants* C', C'' *depending on the optimization problem and the model parameters in Assumption 6 such that:*

(Optimization regret bound) *If* $\|S_t\|_F$ *is bounded, then* $T^{-1}\mathcal{R}(\{X_t\}, \{Z_t\}, T) \le C' \cdot T^{-1/2}$.

(Statistical estimation error) *If Assumptions 1 and 6 hold, and* $\nu \ge \lambda p + \|\Sigma\|_F + 1$, *then for any fixed* $\varepsilon \in (0, 1)$,

$$\|\hat{X}_T - \Pi\|_F \le C'' \left[T^{-1/4} \{\log(1/\varepsilon) + \nu^2\}^{1/2} + T^{-1/2} \log(1/\varepsilon) \{\nu \log(T)\}^{1/2} + \lambda^{1/2} \right]$$

holds with probability at least $1 - \varepsilon$.

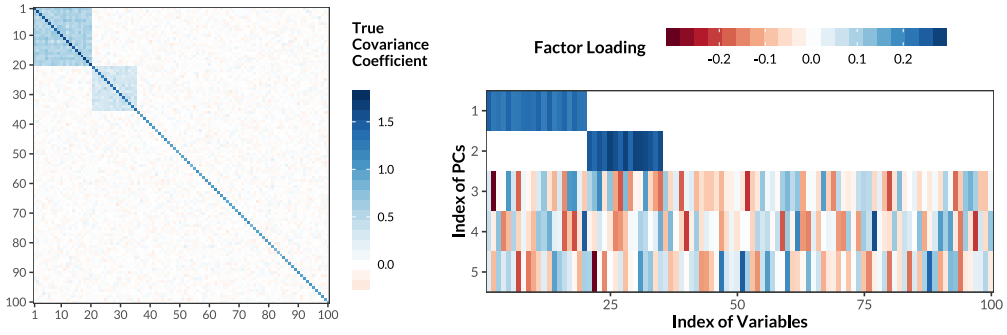
The explicit expressions of C' *and* C'' *are given in Appendix A.2.*

Theorem 4 indicates that λ cannot be too large if the primary goal is to use the final output $\hat{X}_T$ for estimation. Otherwise, a moderate λ leads to more sparse intermediate results and is thus better for interpretation.

5. Simulation Study

5.1. Simulation Setting

In this section we conduct a number of numerical experiments to evaluate the performance of the sparse PCA algorithms proposed in this article. The problem setting is as follows. We assume that the data $Z_1, \ldots, Z_n$ follow independent and identically distributed multivariate normal distribution $N(0, \Sigma)$ in $\mathbb{R}^p$. For online learning algorithms, the data sequence is of infinite length, and the online algorithm will choose a terminal sample size T. The p variables are categorized into three groups: the first signal group contains $d_1 = 20$ variables, the second signal group contains $d_2 = 15$ variables, and the last noise group consists of $(p - d_1 - d_2)$ noise variables. Figure 1(a) gives a visualization of the true covariance matrix Σ with $p = 100$, which shows that most variables are weakly correlated with each other, but the ones within the same signal group have higher correlations. In different experiments, n and p may vary, but d_1 and d_2 are kept fixed.



(a) (b)

Fig. 1: (a) The true covariance matrix Σ with $p = 100$. (b) The eigenvectors of Σ associated with the five largest eigenvalues.

We generate Σ as follows. Let $U_{r_1:r_2,c_1:c_2}$ denote the submatrix of a $p \times p$ matrix U, with row indices r_1 to r_2 and column indices c_1 to c_2. When $r_1 = r_2$ or $c_1 = c_2$, a single index is used. First simulate a U matrix with independent entries such that $U_{1:d_1,1} \sim \mathrm{Unif}(0.9, 1.1)$, $U_{(d_1+1):p,1} = 0$, $U_{1:d_1,2} = 0$, $U_{(d_1+1):(d_1+d_2),2} \sim \mathrm{Unif}(0.9, 1.1)$, $U_{(d_1+d_2+1):p,2} = 0$, and $U_{1:p,3:p} \sim N(0, 1)$. Then a QR decomposition is performed as $U = QR$. Next, let $\Lambda = \mathrm{diag}\{12, 6, \lambda_3, \ldots, \lambda_p\}$, where $\lambda_i \sim \mathrm{Unif}(0, 2)$, and then Σ is computed as $\Sigma = Q\Lambda Q^{\mathrm{T}}$. We design the factor loading matrix in such a way that the first two eigenvectors are sparse, whereas the others are all dense. The nonzero coefficients in the sparse eigenvectors are intentionally made unequal, to avoid cases that are too special. Figure 1(b) shows the first five columns of Q, and clearly the first $d = 2$ columns of Q contain the sparse eigenvectors.

5.2. Batch Algorithms

Since the proposed algorithm (Algorithm 3) solves the same optimization problem as the existing one based on alternating direction method of multipliers (Vu et al., 2013), the focus

of this experiment is to compare their computational efficiency and convergence speed with different sizes of data. Under each pair of (n, p), a data set $Z_1, \ldots, Z_n$ is simulated to compute the sample covariance matrix $S = n^{-1} \sum_{i=1}^{n} Z_i Z_i^{\mathrm{T}}$, and the sparsity parameter is set to $\lambda = 0.5 \cdot \{\log(p)/n\}^{1/2}$. We compute the estimator $\hat{X}$ using both algorithms with initial value $X_0 = V_2 V_2^{\mathrm{T}}$, where V_2 contains the top two eigenvectors of S. For both algorithms, the best step size parameter is chosen by trying ten equally-spaced values ranging from 0.01 to 0.1. In the proposed algorithm we run the subgradient algorithm for $B = 30$ iterations, followed by the proximal-proximal-gradient algorithm towards convergence.

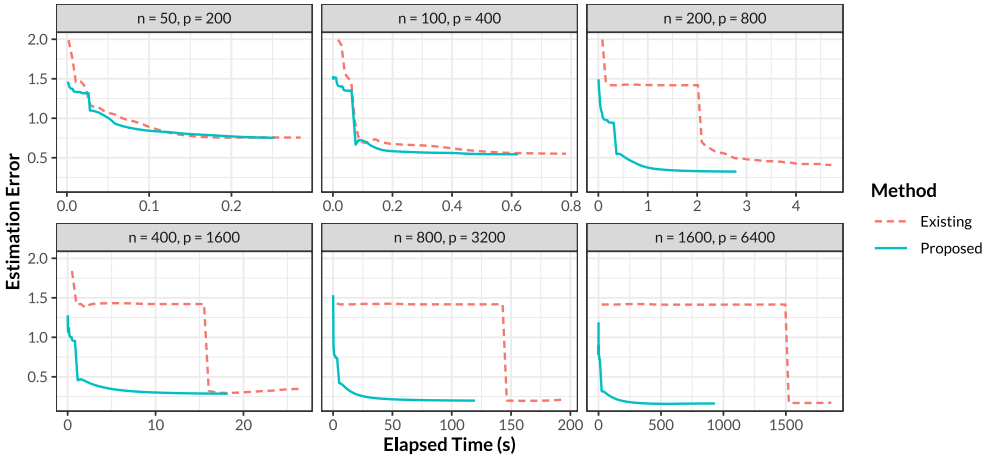


Fig. 2: Comparing the computational efficiency of the existing and proposed algorithms for sparse PCA with convex relaxation. The horizontal axis is the elapsed time in seconds, and the vertical axis stands for the estimation error $\|\hat{X} - \Pi\|_F$.

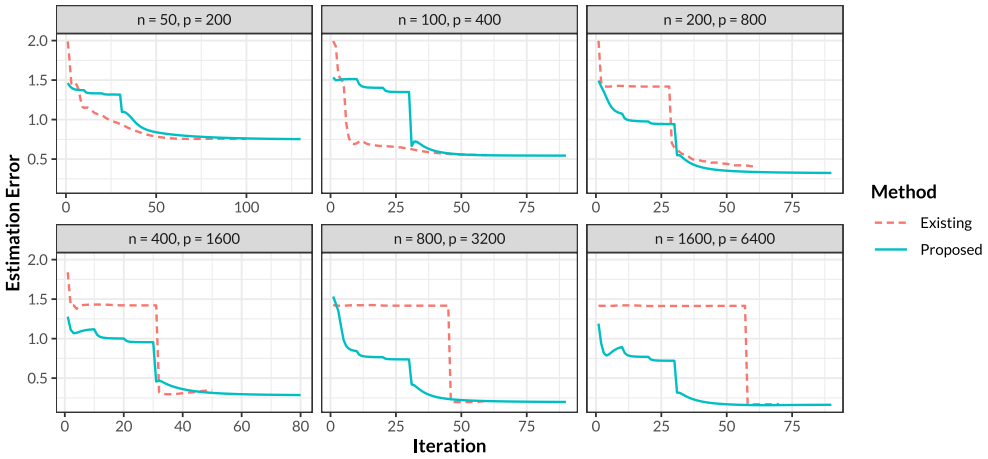


Fig. 3: Comparing the convergence speed of the existing and proposed algorithms for sparse PCA with convex relaxation. The horizontal axis shows the number of iterations of each algorithm.

In Figure 2 and Figure 3, we plot the estimation errors against the computing time and the number of iterations, respectively, showing the following interesting findings. First, as expected,

our algorithm has demonstrated superior computational efficiency compared with the existing one. It is clear in Figure 2 that the curves for the proposed gradient-based algorithm decrease very quickly at early stages of the optimization, which indicates that it is able to provide reasonably accurate solutions from early on. Such a property is crucial, since a common practice for computing sparse PCA is to use convex solutions as good initial values for fast nonconvex methods (Wang et al., 2014; Chen & Wainwright, 2015; Tan et al., 2018). Second, the curves for the existing algorithm have irregular shapes, containing some long "plateaus" and even increasing parts. In practice, such patterns are misleading for convergence tests. In contrast, the curves for our algorithm mostly show a monotone progress after the warm-up stage. Even in the subgradient updates, Figure 3 shows that the estimation error can have significant decrease, compared to the slow convergence of alternating direction method of multipliers at early stages. Finally, even if the same initial value is supplied to both algorithms, the proposed algorithm tends to make better use of it, as the initial errors of our method are smaller than those of the existing algorithm.

5.3. Online Algorithms

The next experiment studies whether the sparsity assumption helps improve the estimation accuracy of PCA in the online learning setting. Specifically, we compare our online gradient-based sparse PCA (Algorithm 4) with a number of well-known online PCA algorithms in the literature, including Oja's stochastic approximation method (Oja & Karhunen, 1985), the incremental PCA method (Arora et al., 2012), and the candid covariance-free incremental PCA (Weng et al., 2003). We fix $T = 200$ and consider two dimension settings $p = 800$ and $p = 1600$, and in each case data points $Z_1, \ldots, Z_T \overset{iid}{\sim} N(0, \Sigma)$ are drawn in a streaming fashion. The step size for online learning algorithms is set to $\alpha_t = 0.1 \cdot t^{-1/2}$, and the sparsity parameter for online sparse PCA is $\lambda = \{\log(p)/n\}^{1/2}$.

For each method, let X_t be the estimator for the true projection matrix $\Pi = \Gamma\Gamma^{\mathrm{T}}$ after receiving the data point Z_t. Figure 4 plots the estimation error $\|X_t - \Pi\|_F$ against t. For each combination of the data generation setting and online learning algorithm, we repeat the experiment ten times, so each panel in Figure 4 shows ten error curves to reflect such variability. It is clear that in the high-dimensional setting, existing online PCA methods have large estimation errors that decay very slowly, whereas our online gradient-based sparse PCA has much better progress.

Moreover, in Figure 5 we visualize the factor loading matrix from the final output of each online learning algorithm, and compare it with the true eigenvectors. As expected, existing online PCA methods have noisy estimates for the factor loading coefficients, making the true signals overwhelmed by the noise. In fact, the incremental PCA and the candid covariance-free incremental PCA fail to detect the second signal group (variable 21 to 35), and Oja's method for the $p = 1600$ case show very weak coefficients for all signal variables (variable 1 to 35). In contrast, the proposed method successfully detects the two signal groups in both settings, and gives near-zero coefficient estimates for most noise variables. These findings further validate the theoretical properties and practical usefulness of the proposed online sparse PCA algorithm.

6. APPLICATION

In this section we apply sparse PCA to an RNA sequencing data set to analyze the co-expression relationship among genes. The aim of our analysis is to detect groups of genes, typically referred to as modules, with high co-expressions. Such an analysis is motivated by the biological conjecture that genes in the same module are likely to be functionally related (Stuart et al., 2003). A computationally efficient sparse PCA algorithm is particularly well suited to this challenging problem for which expression data are available for tens of thousands of genes.

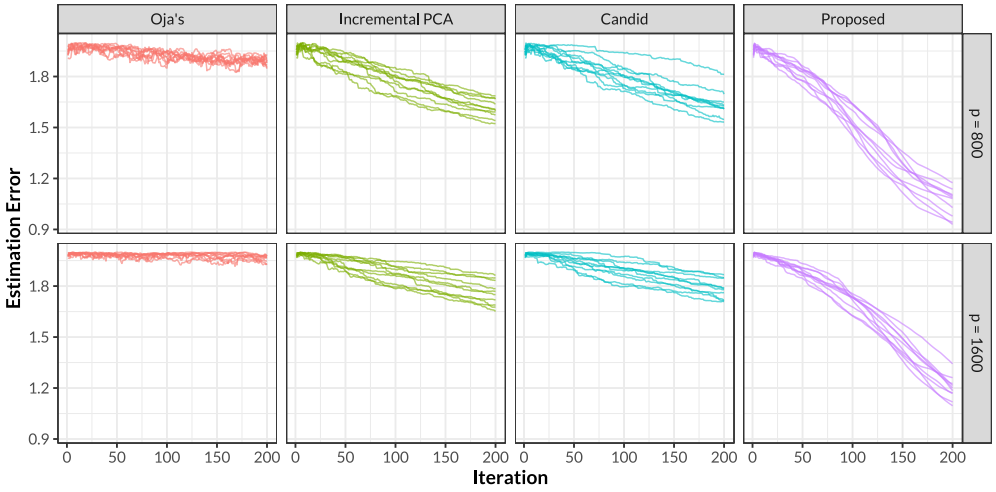


Fig. 4: Estimation errors of various online PCA methods in two data generation settings. Each panel shows ten error curves corresponding to ten independent simulation runs.

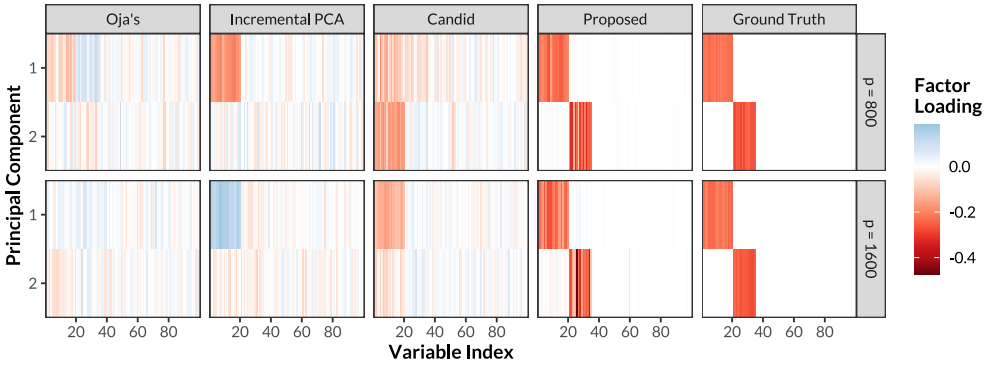


Fig. 5: Factor loadings of the first two principal components (showing the first 100 variables) estimated by different online learning algorithms.

We study the brain gene expression data collected by the CommonMind Consortium, which contain $p = 16,423$ genes from 258 schizophrenia subjects and 279 control subjects (Fromer et al., 2016). Such a dimensionality makes the existing algorithm infeasible on an average computer. The control group is used as a baseline, and our main interest is in the schizophrenia group. We compute Pearson's correlation coefficients between genes using the processed and normalized expression data provided by the CommonMind Consortium, and then apply sparse PCA to the sample correlation matrix. The number of sparse principal components is chosen to be $d = 5$, and the sparsity parameter λ is selected in the following way. First, we compute the solution paths of sparse PCA in both groups based on a common sequence of λ values. Then for each λ, two active sets $\Omega^{\lambda}_{ctr}, \Omega^{\lambda}_{scz} \subset \{1, 2, \ldots, p\}$ are determined, where $i \in \Omega^{\lambda}_{ctr}$ if the ith gene in the control group has at least one nonzero factor loading in the five sparse principal components, and $i \in \Omega^{\lambda}_{scz}$ is defined likewise. We limit the range of λ so that $\min\{|\Omega^{\lambda}_{ctr}|, |\Omega^{\lambda}_{scz}|\} \geq 50$

and $\max\{|\Omega^{\lambda}_{ctr}|, |\Omega^{\lambda}_{scz}|\} \leq 300$, where $|\Omega|$ denotes the cardinality of a set Ω. Define the overlapping coefficient as $V(\lambda) = |\Omega^{\lambda}_{ctr} \cap \Omega^{\lambda}_{scz}|/|\Omega^{\lambda}_{ctr} \cup \Omega^{\lambda}_{scz}|$, and λ is chosen to maximize $V(\lambda)$, indicating that these two groups share maximal common structures. Using this approach, we select $\lambda = 0.85$, under which $|\Omega_{ctr}| = 292$, $|\Omega_{scz}| = 185$, and $|\Omega_{ctr} \cap \Omega_{scz}| = 114$.

After computing the sparse PCA solution for the schizophrenia group at the selected λ, the genes in the active set are clustered based on their factor loadings, with the number of clusters set to $k = 5$. For display, the indices of genes are reordered so that the genes in the same cluster are adjacent. Figure 6 shows the sample correlation matrix and factor loadings based on the reordered indices of selected genes.

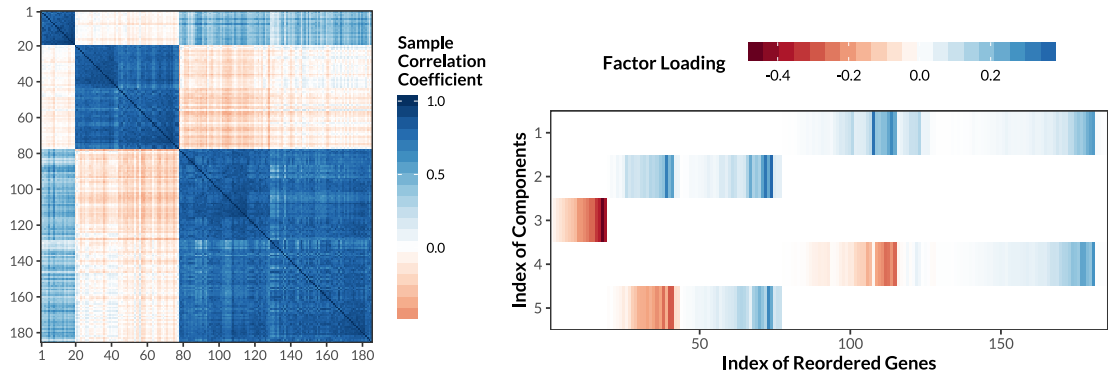


Fig. 6: The reordered sample correlation matrix of the selected genes in the schizophrenia group (left) and the reordered factor loadings (right).

It can be easily observed from Figure 6 that there are three major modules in the correlation matrix, and the second and third modules have two sub-modules, respectively, resulting in five clusters in total. Such a structure is clearly reflected in the factor loadings, in which the first three components define the major modules, whereas the last two components add sub-structures to the second and third modules.

To validate our results, we compare the clusters reflected in Figure 6 with the modules published in the literature. Table 1 demonstrates the cross table for the two methods of module assignment on the selected genes, where the numbered modules are given by our approach, and the ones labeled by color names are the results provided by Fromer et al. (2016), using the weighted gene co-expression network analysis (Zhang & Horvath, 2005). It is clear that our modules are well aligned with the published ones, with three extra advantages. First, our clusters have smaller sizes and stronger within-group correlation. For instance, the Green module contains 414 genes, whereas our M-1, a subset of the Green module, has only 19 genes. In many studies, researchers are more interested in a small number of genes that are representative for the whole module. Second, we have detected highly correlated genes that are assigned to different published modules. As an example, the two genes in the Tan module are highly correlated with other M-4 genes (a subset of Turquoise), with average sample correlation coefficients 0.817 and 0.794, respectively. Finally, our clusters have revealed sub-structures within large modules, for example M-2 and M-3 are sub-modules of Brown.

Next, by comparing with the control group, we study the structural change of gene co-expression relationship in the schizophrenia group. Consider the genes that are selected in the schizophrenia group but not in the control group, forming the gene set $\Omega^{U}_{scz} = \Omega_{scz} \backslash \Omega_{ctr}$. Figure

Table 1: Cross table for sparse-PCA-based modules (row) and the published modules (column). The numbers in the parentheses stand for the sizes of the published modules.

	Green (414)	Brown (528)	Turquoise (1155)	Tan (248)	Blue (609)
M-1	19	0	0	0	0
M-2	0	24	0	0	0
M-3	0	34	0	0	0
M-4	0	0	49	2	0
M-5	0	0	53	0	4

7 illustrates the sample correlation matrices on Ω^U_{scz} for both the control group (left panel) and the schizophrenia group (middle panel). In addition, to better visualize the correlation pattern, density curves of off-diagonal correlation coefficients are shown in the right panel of Figure 7.

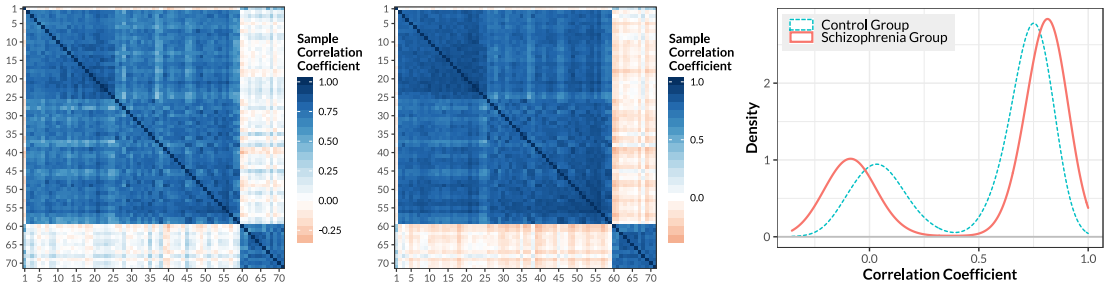


Fig. 7: Comparison of correlation matrices on schizophrenia-group-specific genes Ω^U_{scz}. Left: the correlation matrix on Ω^U_{scz} for the control group. Middle: the correlation matrix for the schizophrenia group. Right: density curves for the off-diagonal correlation coefficients.

Figure 7 highlights an interesting difference between the control group and the schizophrenia group. In both groups, the correlation matrices indicate a similar two-block structure, but density curves of the correlations summarize the differences between groups. Both exhibit two modes, representing the between-module and within-module correlation coefficients, respectively; however, the coefficients in the schizophrenia group are obviously more extreme than those in the control group. The first mode differs in sign, indicating that the small positive between-module correlations in the control group are largely negative in the schizophrenia group. These findings provide insights for future studies of schizophrenia based on brain gene expression data.

7. DISCUSSION

The framework used in our analysis has a great potential for further extensions. First, within the sparse PCA framework, the efficient algorithm can be developed for other types of problems that come with a different convex penalty term, such as the trend filtering (Tibshirani, 2014) or the localized functional PCA (Chen & Lei, 2015). Second, the two technical tools developed in this article, the gradient-based and projection-free optimization method for highly constrained problems, and the analysis of online learning algorithms, can be extended to other interesting

statistical models. An example is the graphical lasso (Friedman et al., 2008), where the precision matrix is constrained in the positive semidefinite cone with an entry-wise ℓ_1 penalty. Similar to sparse PCA, online learning algorithms may be developed for the graphical lasso using a trivially constrained formulation of the objective function.

ACKNOWLEDGMENTS

This work was supported by NIMH grants R37MH057881-22, R37MH057881-22S, and R01MH123184, and NSF grants DMS-1553884 and DMS-2015492.

Data were generated as part of the CommonMind Consortium supported by funding from Takeda Pharmaceuticals Company Limited, F. Hoffman-La Roche Ltd and NIH grants R01MH085542, R01MH093725, P50MH066392, P50MH080405, R01MH097276, RO1-MH-075916, P50M096891, P50MH084053S1, R37MH057881, AG02219, AG05138, MH06692, R01MH110921, R01MH109677, R01MH109897, U01MH103392, and contract HHSN271201300031C through IRP NIMH. Brain tissue for the study was obtained from the following brain bank collections: the Mount Sinai NIH Brain and Tissue Repository, the University of Pennsylvania Alzheimer's Disease Core Center, the University of Pittsburgh NeuroBioBank and Brain and Tissue Repositories, and the NIMH Human Brain Collection Core. CMC Leadership: Panos Roussos, Joseph Buxbaum, Andrew Chess, Schahram Akbarian, Vahram Haroutunian (Icahn School of Medicine at Mount Sinai), Bernie Devlin, David Lewis (University of Pittsburgh), Raquel Gur, Chang-Gyu Hahn (University of Pennsylvania), Enrico Domenici (University of Trento), Mette A. Peters, Solveig Sieberts (Sage Bionetworks), Thomas Lehner, Stefano Marenco, Barbara K. Lipska (NIMH).

A. APPENDIX

A.1. Conservativeness of μ in Corollary 1

In this section we use an example to demonstrate that the bound for μ developed in Corollary 1 may be conservative, and in practice a smaller value for μ can be used that still results in the correct solution. Consider the experiment in Section 5.2 with $n = 50$ and $p = 200$, and set $\mu_{\max} = (\sqrt{2}+1)(L+1)\{p/(d+1)\}^{1/2}$. Corollary 1 shows that any $\mu \geq \mu_{\max}$ guarantees the equivalence between $\min_{X\in\mathcal{K}} f(X)$ and $\min_{X\in\mathcal{X}} \mathcal{L}(X)$, and we denote by $\hat{X}_*$ the solution to $\min_{X\in\mathcal{K}} f(X)$.

Then we test different values of μ by solving $\hat{X} = \arg\min_{X\in\mathcal{X}} \mathcal{L}(X)$, where $\mathcal{L}(X)$ and $\hat{X}$ implicitly depend on μ. Figure 8 shows the relation between $\|\hat{X} - \hat{X}_*\|_F$ and μ, and it is clear that $0.05\mu_{\max}$ suffices to produce the correct solution.

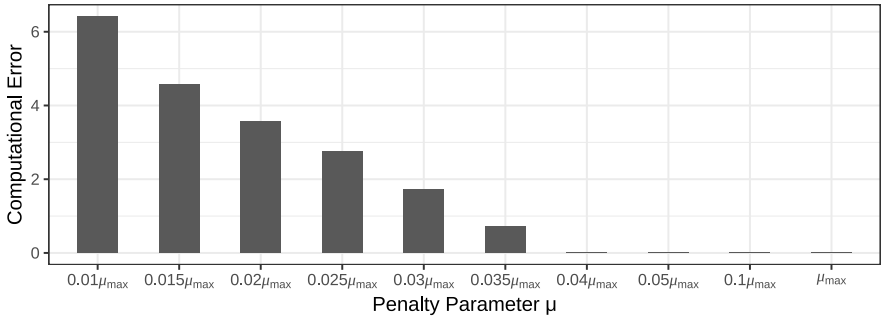


Fig. 8: The computational error $\|\hat{X} - \hat{X}_*\|_F$ when different μ is used in (5).

A.2. *Expressions for constants and bounds*

Theorem 3: The constant is $C = \max\{\alpha^{-1}(C_0^2 + 4C_0L_g), 2C_0L_g\} + 2C_0L_g$, where

$$L_g^2 = (\lambda p)^2 + \left[\|S\|_F + \mu\left\{1 + (p+d)^{1/2}(d+1)^{1/2}\right\}\right]^2,$$

and $C_0 > 0$ is a constant that only depends on X_0 and the optimal point of the optimization problem.

Theorem 4: The regret bound in explicit form is given by

$$\frac{1}{T}\mathcal{R}(\{X_t\}, \{Z_t\}, T) \le \frac{2d/\alpha_0 + \alpha_0 C_2}{T^{1/2}} + \frac{\alpha_0}{2T}\sum_{t=1}^{T}\frac{\|S_{t+1}\|_F^2 + C_1\|S_{t+1}\|_F}{t^{1/2}}, \tag{9}$$

and the estimation error bound is $\|\hat{X}_T - \Pi\|_F \le C(T) + (2/\delta_d)^{1/2} \cdot \{\lambda s d^{1/2} + C(T)\}^{1/2}$, where $C(T) = C_3/T^{1/2} + C_4\ell(T)/T + C_5\{\ell(T)\}^{2/3}/T$ and $\ell(T) = \log(T) + 1$. The relevant quantities are

$$\begin{aligned}
C_1 &= \lambda p + \nu_0 p^{1/2}\{(p+d)^{1/2} + (d+1)^{-1/2}\},\\
C_2 &= \nu_0^2 p(p+d) + 2(\lambda p)^2 + 2\lambda p\nu_0\{p(p+d)\}^{1/2} + 2\nu_0\{p/(d+1)\}^{1/2}C_1,\\
C_3 &= 2d/\alpha_0 + \alpha_0\{C_2 + 2\|R\|^4(p^2+16p) + C_1\|R\|^2 p\} + D_1,\\
C_4 &= \alpha_0 C_1\|R\|^2 D_2/2,\\
C_5 &= \alpha_0\|R\|^4 D_3,\\
D_1 &= \max\left\{C_\eta\|R\| \cdot \|R\|_F(d^{1/2}+1)(2\varepsilon_l)^{1/2},\ 2c_\eta\|R\|^2(d^{1/2}+1)\varepsilon_l/T^{1/2}\right\},\\
D_2 &= 8 \cdot \max\left\{\varepsilon_l/\ell(T),\ \{p\varepsilon_l/\ell(T)\}^{1/2}\right\},\\
D_3 &= 600p \cdot \max\left\{C_\zeta^{1/2}\{\varepsilon_l + \log(2)\}^{1/2}\{\ell(T)\}^{-1/6},\ C_\zeta^2\{\varepsilon_l + \log(2)\}^2\{\ell(T)\}^{-2/3}\right\},
\end{aligned}$$

where $\nu_0 = (\sqrt{2}+1)\nu$, $\varepsilon_l = \log(3/\varepsilon)$, and C_η, c_η, C_ζ are positive absolute constants.

A.3. *Computation of* $\mathrm{prox}_{\alpha f_2}(X)$

By definition $\mathrm{prox}_{\alpha f_2}(X) = \arg\min_{U\in\mathcal{X}}\left\{f_2(U) + (2\alpha)^{-1}\|U - X\|_F^2\right\}$. If $Y = X + \alpha S$ has the eigen-decomposition $Y = \sum_{i=1}^p \theta_i\gamma_i\gamma_i^{\mathrm{T}}$, then it can be verified that $\mathrm{prox}_{\alpha f_2}(X) = \sum_{i=1}^p u_i\gamma_i\gamma_i^{\mathrm{T}}$, where $u = (u_1, \ldots, u_p)^{\mathrm{T}}$ is the solution to the vector optimization problem

$$\min_u\ \frac{1}{2\alpha}\|u - \theta\|^2 + \mu p^{-1/2}\left|\sum_{i=1}^p u_i - d\right| + \mu r_1[\max(u) - 1]_+ + \alpha\mu r_2[-\min(u)]_+. \tag{10}$$

Introduce auxiliary variables $v_1 = \max(u)$, $v_2 = [v_1 - 1]_+$, $v_3 = \min(u)$, $v_4 = [-v_3]_+$, and $v_5 = |\sum_{i=1}^p u_i - d|$. Then (10) reduces to a quadratic programming problem

$$\begin{aligned}
&\min_{u,v}\ \frac{1}{2\alpha}\|u - \theta\|^2 + \mu r_1 v_2 + \mu r_2 v_4 + \mu p^{-1/2} v_5\\
&\text{s.t. } v_3 \le u_i \le v_1, v_2 \ge 0, v_2 \ge v_1 - 1, v_4 \ge 0, v_4 \ge -v_3, v_5 \ge \textstyle\sum_{i=1}^p u_i - d, v_5 \ge d - \sum_{i=1}^p u_i,
\end{aligned}$$

which readily has efficient solvers.

Next we show that u can actually be obtained without computing the full eigen-decomposition of Y. First, u must be ordered, $u_1 \ge \cdots \ge u_p$, as θ is ordered. Second, since r_2 is free to choose as long as $r_2 \ge \{p(d+1)\}^{1/2}$, we intentionally set $r_2 = +\infty$, which enforces the condition $u_i \ge 0$. Finally, combined with the penalty term $|\sum_{i=1}^p u_i - d|$, we find that u is a sparse vector with the first few elements being positive and all the rest being zero. Then u can be computed in the following way. Define the sub-vector $\theta_{1:I} = (\theta_1, \ldots, \theta_I)^{\mathrm{T}}$, and denote by $u_{1:I}$ the solution to (10) with θ replaced by $\theta_{1:I}$. We test values $I = 1, 2, \ldots$ until $u_I = 0$, and it is guaranteed that $u_i = 0$ for $i \ge I$, leading to $\mathrm{prox}_{\alpha f_2}(X) = \sum_{i=1}^{I-1} u_i\gamma_i\gamma_i^{\mathrm{T}}$. In other words, we only compute I eigen-pairs of Y instead of p, and most of the time I is just slightly larger than d. In our actual implementation, we use the implicitly restarted Lanczos method (Sorensen, 1997) to compute eigenvalues and eigenvectors. Finally, the cost for the quadratic programming problems can be ignored compred to the eigenvalue computation.

REFERENCES

ABERNETHY, J., BARTLETT, P. L., RAKHLIN, A. & TEWARI, A. (2008). Optimal strategies and minimax lower bounds for online convex games. In *Conference on Learning Theory*.

ARORA, R., COTTER, A., LIVESCU, K. & SREBRO, N. (2012). Stochastic optimization for pca and pls. In *50th Annual Allerton Conference on Communication, Control, and Computing*.

BERTSEKAS, D. P. (2011). Incremental proximal methods for large scale convex optimization. *Mathematical programming* **129**, 163.

BOYD, S., PARIKH, N., CHU, E., PELEATO, B. & ECKSTEIN, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning* **3**, 1–122.

CHEN, K. & LEI, J. (2015). Localized functional principal component analysis. *Journal of the American Statistical Association* **110**, 1266–1275.

CHEN, Y. & WAINWRIGHT, M. J. (2015). Fast low-rank estimation by projected gradient descent: general statistical and algorithmic guarantees. *arXiv preprint arXiv:1509.03025* .

D'ASPREMONT, A., GHAOUI, L. E., JORDAN, M. I. & LANCKRIET, G. R. (2005). A direct formulation for sparse pca using semidefinite programming. In *Advances in Neural Information Processing Systems*.

DUCHI, J., HAZAN, E. & SINGER, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* **12**, 2121–2159.

FRIEDMAN, J., HASTIE, T. & TIBSHIRANI, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **9**, 432–441.

FROMER, M., ROUSSOS, P., SIEBERTS, S. K., JOHNSON, J. S., KAVANAGH, D. H., PERUMAL, T. M., RUDERFER, D. M., OH, E. C., TOPOL, A., SHAH, H. R. et al. (2016). Gene expression elucidates functional impact of polygenic risk for schizophrenia. *Nature Neuroscience* **19**, 1442.

GAJJAR, S., KULAHCI, M. & PALAZOGLU, A. (2018). Real-time fault detection and diagnosis using sparse principal component analysis. *Journal of Process Control* **67**, 112–128.

GRBOVIC, M., LI, W., XU, P., USADI, A. K., SONG, L. & VUCETIC, S. (2012). Decentralized fault detection and diagnosis via sparse pca based decomposition and maximum entropy decision fusion. *Journal of Process Control* **22**, 738–750.

HOTELLING, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology* **24**, 417.

JOHNSTONE, I. M. & LU, A. Y. (2009). On consistency and sparsity for principal components analysis in high dimensions. *Journal of the American Statistical Association* **104**, 682–693.

JOLLIFFE, I. T. (2002). *Principal component analysis*. Springer Series in Statistics. Springer New York, NY.

JOLLIFFE, I. T., TRENDAFILOV, N. T. & UDDIN, M. (2003). A modified principal component technique based on the lasso. *Journal of Computational and Graphical Statistics* **12**, 531–547.

JOURNÉE, M., NESTEROV, Y., RICHTÁRIK, P. & SEPULCHRE, R. (2010). Generalized power method for sparse principal component analysis. *Journal of Machine Learning Research* **11**, 517–553.

JUNG, S. & MARRON, J. S. (2009). Pca consistency in high dimension, low sample size context. *The Annals of Statistics* **37**, 4104–4130.

KINGMA, D. P. & BA, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

KUNDU, A., BACH, F. & BHATTACHARYA, C. (2018). Convex optimization over intersection of simple sets: improved convergence rate guarantees via an exact penalty approach. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*.

LEE, S., EPSTEIN, M. P., DUNCAN, R. & LIN, X. (2012). Sparse principal component analysis for identifying ancestry-informative markers in genome-wide association studies. *Genetic Epidemiology* **36**, 293–302.

LEI, J. & VU, V. Q. (2015). Sparsistency and agnostic inference in sparse pca. *The Annals of Statistics* **43**, 299–322.

LEPSKII, O. (1991). On a problem of adaptive estimation in gaussian white noise. *Theory of Probability & Its Applications* **35**, 454–466.

LI, C. J., WANG, M., LIU, H. & ZHANG, T. (2018). Near-optimal stochastic approximation for online principal component estimation. *Mathematical Programming* **167**, 75–97.

LUO, L., XIONG, Y., LIU, Y. & SUN, X. (2019). Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations*.

MA, Z. (2013). Sparse principal component analysis and iterative thresholding. *The Annals of Statistics* **41**, 772–801.

MAHDAVI, M., YANG, T., JIN, R., ZHU, S. & YI, J. (2012). Stochastic gradient descent with only one projection. In *Advances in Neural Information Processing Systems*.

MARINOV, T. V., MIANJY, P. & ARORA, R. (2018). Streaming principal component analysis in noisy settings. In *35th International Conference on Machine Learning*.

OJA, E. & KARHUNEN, J. (1985). On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of Mathematical Analysis and Applications* **106**, 69–84.

PEARSON, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **2**, 559–572.

RYU, E. K. & YIN, W. (2019). Proximal-proximal-gradient method. *Journal of Computational Mathematics* **37**, 778–812.

SHE, Y. (2017). Selective factor extraction in high dimensions. *Biometrika* **104**, 97–110.

SHEN, H. & HUANG, J. Z. (2008). Sparse principal component analysis via regularized low rank matrix approximation. *Journal of Multivariate Analysis* **99**, 1015–1034.

SORENSEN, D. C. (1997). Implicitly restarted arnoldi/lanczos methods for large scale eigenvalue calculations. In *Parallel Numerical Algorithms*. Springer, pp. 119–165.

STUART, J. M., SEGAL, E., KOLLER, D. & KIM, S. K. (2003). A gene-coexpression network for global discovery of conserved genetic modules. *Science* **302**, 249–255.

TAN, K. M., WANG, Z., LIU, H. & ZHANG, T. (2018). Sparse generalized eigenvalue problem: optimal statistical rates via truncated rayleigh flow. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **80**, 1057–1086.

TIBSHIRANI, R. J. (2014). Adaptive piecewise polynomial estimation via trend filtering. *The Annals of Statistics* **42**, 285–323.

VU, V. Q., CHO, J., LEI, J. & ROHE, K. (2013). Fantope projection and selection: a near-optimal convex relaxation of sparse pca. In *Advances in Neural Information Processing Systems*.

VU, V. Q. & LEI, J. (2013). Minimax sparse principal subspace estimation in high dimensions. *The Annals of Statistics* **41**, 2905–2947.

WANG, C. & LU, Y. M. (2016). Online learning for sparse pca in high dimensions: exact dynamics and phase transitions. In *2016 IEEE Information Theory Workshop*.

WANG, Z., LU, H. & LIU, H. (2014). Nonconvex statistical optimization: minimax-optimal sparse pca in polynomial time. *arXiv preprint arXiv:1408.5352* .

WARMUTH, M. K. & KUZMIN, D. (2008). Randomized online pca algorithms with regret bounds that are logarithmic in the dimension. *Journal of Machine Learning Research* **9**, 2287–2320.

WARSA, J. S., WAREING, T. A., MOREL, J. E., MCGHEE, J. M. & LEHOUCQ, R. B. (2004). Krylov subspace iterations for deterministic k-eigenvalue calculations. *Nuclear Science and Engineering* **147**, 26–42.

WENG, J., ZHANG, Y. & HWANG, W.-S. (2003). Candid covariance-free incremental principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**, 1034–1040.

WITTEN, D. M., TIBSHIRANI, R. & HASTIE, T. (2009). A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics* **10**, 515–534.

YANG, T., LIN, Q. & ZHANG, L. (2017). A richer theory of convex constrained optimization with reduced projections and improved rates. In *34th International Conference on Machine Learning*.

YANG, W. & XU, H. (2015). Streaming sparse principal component analysis. In *32nd International Conference on Machine Learning*.

ZEILER, M. D. (2012). Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701* .

ZHANG, B. & HORVATH, S. (2005). A general framework for weighted gene co-expression network analysis. *Statistical Applications in Genetics and Molecular Biology* **4**.

ZHANG, Y. & GHAOUI, L. E. (2011). Large-scale sparse principal component analysis with application to text data. In *Advances in Neural Information Processing Systems*.

ZHU, L., LEI, J., DEVLIN, B. & ROEDER, K. (2017). Testing high-dimensional covariance matrices, with application to detecting schizophrenia risk genes. *The Annals of Applied Statistics* **11**, 1810.

ZOU, H., HASTIE, T. & TIBSHIRANI, R. (2006). Sparse principal component analysis. *Journal of Computational and Graphical Statistics* **15**, 265–286.

ZOU, H. & XUE, L. (2018). A selective overview of sparse principal component analysis. *Proceedings of the IEEE* **106**, 1311–1320.